

SQL (Structured Query Language) & Data Modelling/Database Design

About Me (Guruprasad Veerannavaru)

- From Karnataka, currently working in Bangalore
- B.E in Computer Science from NIE Mysore – 2021
- CBSE 12th from Sainik School Bijapur, Karnataka
- Active Quizzer
 - Winner Tata Crucibles (Business Quiz) Mysore Zonal
 - Runner-up's in TCS Tech Bytes Karnataka
- AIR 4 – UPSC NDA (National Defense Academy)
- Hobbies – Reading Books, Travelling, Running

Experience



Software Engineer Intern

Microsoft · Internship

May 2020 - Jul 2020 · 3 mos

Bengaluru, Karnataka



Full Stack Developer Intern

Clinikk Healthcare · Internship

Aug 2020 - Nov 2020 · 4 mos

Bengaluru, Karnataka, India



Microsoft

2 yrs 1 mo

Bengaluru, Karnataka, India



Software Engineer

Full-time

Aug 2021 - Present · 1 yr 6 mos



Software Engineer Intern

Internship

Jan 2021 - Jul 2021 · 7 mos



Mentor

Newton School

Oct 2021 - Present · 1 yr 4 mos



<https://www.linkedin.com/in/urguru>



<https://www.youtube.com/c/urguru>



<https://github.com/urguru>



https://twitter.com/urguru_4648

Course Overview

- Introduction to SQL
 - Databases, Tables
 - Classification of SQL – DDL, DML, DCL, TCL
 - DDL – CREATE, ALTER, DROP
 - DML – SELECT, INSERT, UPDATE, DELETE
 - DCL – GRANT, REVOKE
 - TCL – COMMIT, ROLLBACK, SAVEPOINT
 - Data types, Operators
 - Keys – Primary, Foreign, Composite, Unique, Alternate
 - Integrity Constraints – Domain Integrity Constraints, Entity Integrity Constraints, Referential Integrity Constraints
 - Joins – Outer Joins, Left Outer Joins, Right Outer Joins, Inner Joins.
 - Queries, Subqueries, Functions, Flow Control (IF, CASE, WHILE, FOR, LOOP), Stored routines
 - Views
 - Concurrency and locking (Implicit locks, explicit locks, row level locks, table level locks, database level locks)
 - Indexes, Cursors, Triggers, Events
 - Tuning SQL queries and optimizing performance
 - SQL Databases vs NoSQL Databases
 - ACID, CAP
 - How SQL databases internally works

Course Overview

- Data modelling and database design
 - What is Data Model – (Steps Involved, Conceptual Data Modelling, Database design)
 - Entity Relationship model
 - Elements of entity relationship model
 - Entities, Subtypes, supertypes
 - Regular, Strong, Weak entities
 - Identifying and modelling entities
 - Relationships – Modelling, defining and types of relationships
 - Minimum and maximum relationships
 - Attributes- Finding attributes, meaningful components for attributes
 - Diagrammatic conventions
 - Transforming entity relationship model into relational schema
 - Logical database design (Top-Down Approach-E/R Modelling, Bottom-Up Approach-Normalization)
 - Difference between top-down and bottom-up approach
 - Normalization
 - Need for normalization
 - Forms of normalization – 1nf,2nf,3nf, Boyce Code Normalization
 - Case studies and discussion
 - Designing few real life databases



The story of a library

- DBMS (Database management system) – Allows the bookstore owner to store books inventory – Easier CRUD (Create, Read, Update, Delete)
- SQL (Structured Query Language) – Language used to communicate with DBMS and retrieve the information stored in database

Real life uses of DBMS

- Banking
- Retail
- Healthcare
- Transportation
- Telecommunications
- E-commerce
- Social Media
- Government


Different types of DBMS

- Relational DBMS (RDBMS): MySQL, Oracle Database, Microsoft SQL Server, PostgreSQL, etc.
- Hierarchical DBMS: IBM's Information Management System (IMS)
- Network DBMS: Integrated Data Store (IDS)
- Object-oriented DBMS (OODBMS): MongoDB, OrientDB, etc.
- Document DBMS: MongoDB, Couchbase, etc.
- Columnar DBMS: Apache Cassandra, Hbase, etc.
- NoSQL DBMS: MongoDB, Redis, Riak, Cassandra, etc.
- In-memory DBMS: SAP HANA, Oracle TimesTen, Microsoft SQL Server Hekaton, etc.

Relational DBMS

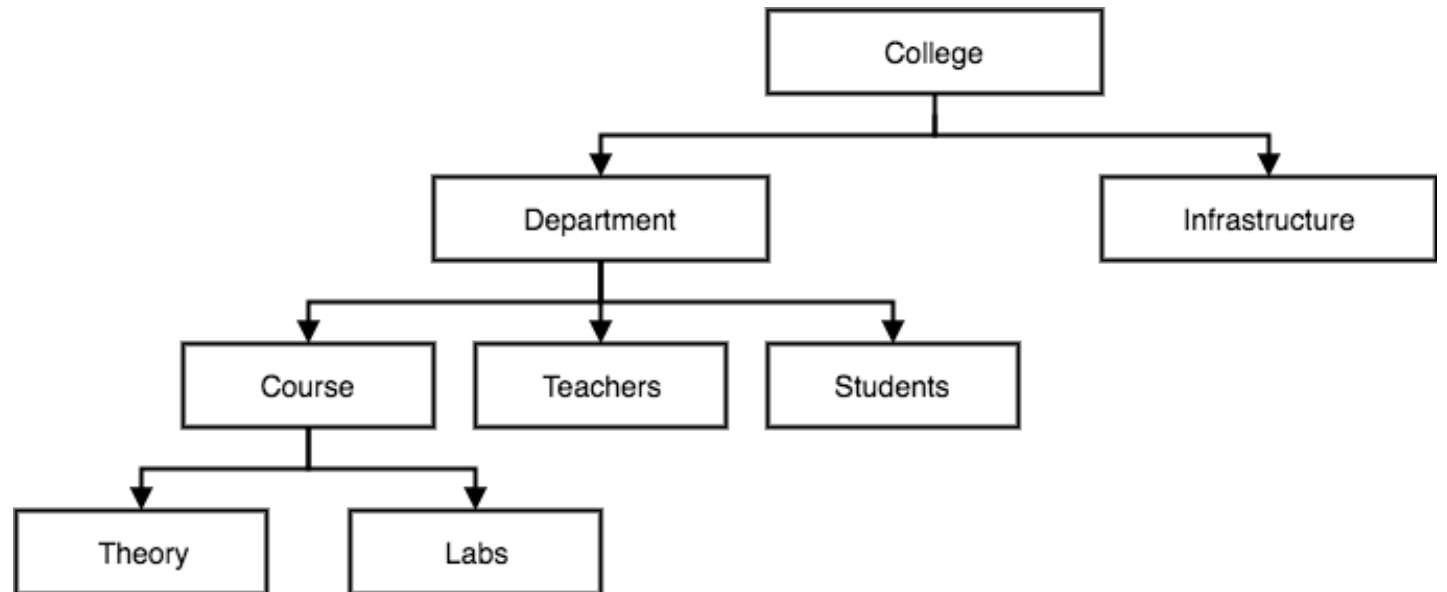
student_id	name	age
1	Akon	17
2	Bkon	18
3	Ckon	17
4	Dkon	18

subject_id	name	teacher
1	Java	Mr. J
2	C++	Miss C
3	C#	Mr. C Hash
4	Php	Mr. P H P

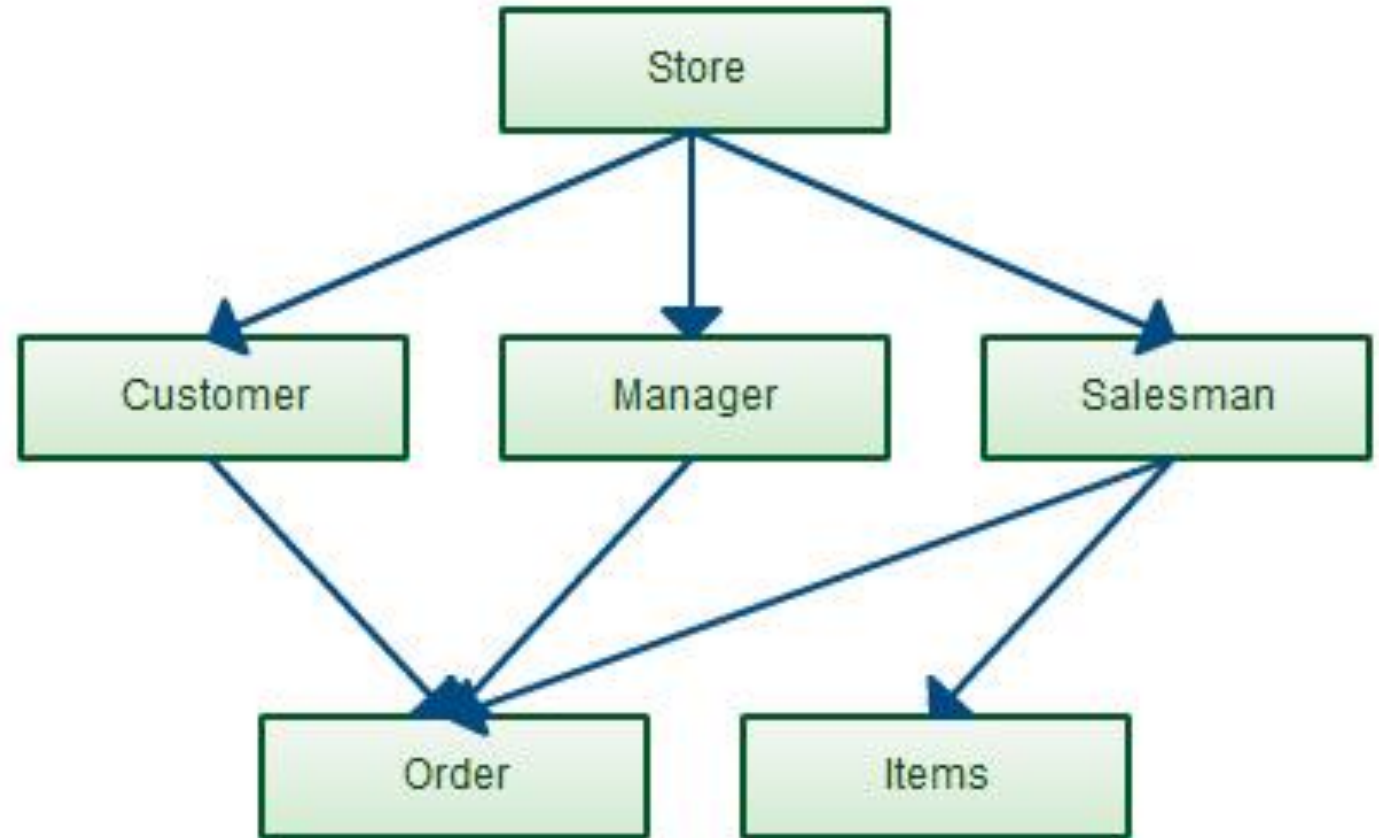


student_id	subject_id	marks
1	1	98
1	2	78
2	1	76
3	2	88

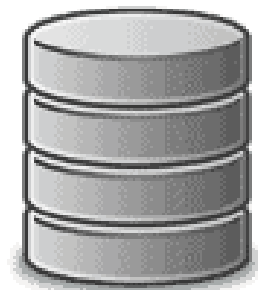
Hierarchical DBMS



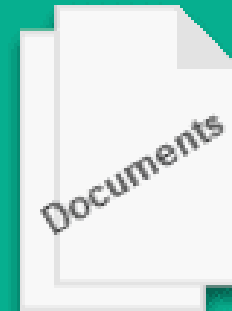
Network
DBMS



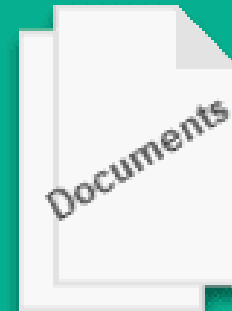
Document
DBMS



Collection 1



Collection 2



Columnar DBMS

row-store



- + easy to add/modify a record
- might read in unnecessary data

column-store



- + only need to read in relevant data
- tuple writes require multiple accesses

=> suitable for read-mostly, read-intensive, large data repositories

Our focus
Relational Database Management Systems

Requirements for a DBMS to be an RDBMS

- Data Integrity – Data stored is accurate and consistent
- Data Security – Must provide mechanisms for data access and control
- Data Concurrency – Must allow multiple concurrent access
- Data recovery – Mechanisms to recover data in case of failure/system crash
- Data Independence – Separation of logical structure from physical storage layer
- SQL Support – RDBMS should support SQL
- ACID Support – Atomicity, Consistency, Isolation, Durability
- Scalability – Large amount of data and users
- Indexing
- Performance – Query execution, Data retrieval

SQL – Structured Query Language

- Standard language used to interact with RDBMS
- ANSI Standard – 1986

Purpose of SQL Standard

- Specify syntax/semantics for data definition and manipulation
- Define data structures and basic operations
- Enable portability of database definition and application modules
- Specify minimal (level 1) and complete (level 2) standards
- Allow for later growth/enhancement to standard (referential integrity, transaction management, user-defined functions, extended join operations, national character sets)

Benefits of using SQL

- Reduced training costs
- Productivity
- Application portability
- Application longevity
- Reduced dependence on a single vendor
- Cross-system communication

Database

- Collection of related
 - Tables
 - Views
 - Stored Procedures
 - Functions
 - Triggers
 - Events
 - Indexes
 - Etc...

Table OR Relation

- Holds all the information or data
- Primary Key – Unique Column
- Column OR Attribute
- Tuple OR Row
- Degree – Number of columns
- Cardinality – Number of rows
- Domain – A data type with optional constraints

Table

Table also called Relation

Primary Key

Domain
Ex: NOT NULL

© guru99.com

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive

Tuple OR Row

Total # of rows is **Cardinality**

Column OR Attributes

Total # of column is **Degree**

Classification of SQL - DDL – Data definition language

- Used to define the following for tables, views, indexes, and other database objects
 - Structure of a database
 - Creation
 - Modification
 - Deletion.
- Example
 - CREATE TABLE
 - ALTER TABLE
 - DROP TABLE
 - CREATE INDEX
 - DROP INDEX
- Executed by Database Administrators (DBAs) or developers
- Usually need specific privileges to execute
- Executed infrequently as compared to DML (Data Manipulation Language) statements which are executed more frequently.

Classification of SQL - DML – Data Manipulation Language

- Used to manipulate the data stored in a database
- Include SELECT,INSERT,UPDATE,DELETE
- Executed more frequently than DDL

Classification of SQL - DCL – Data Control Language

- Used to control access to a database and its objects
- Manage the security of a database by controlling who can access the data and what actions they can perform
- Examples – GRANT, REVOKE, CREATE ROLE
 - GRANT SELECT ON employees TO public;
 - REVOKE SELECT ON employees FROM public;
 - CREATE ROLE data_analyst;
 - SET ROLE data_analyst;
- Usually executed by DBAs
- Critical to enforce security and compliance

Classification of SQL - TCL –Transaction Control Language

- Used to manage changes a database made by transaction
- Transaction
 - Sequence of DML commands that are executed as single unit
 - Example –
 - Deduct 5 rupees from account A and add them to account B.
- Used to ensure that data remains in consistent in case of failure
- Keywords – BEGIN TRANSACTION, COMMIT, ROLLBACK, SAVEPOINT, ROLLBACK TO
- Used to maintain integrity and consistency of data
- Examples
 - BEGIN TRANSACTION;
 - COMMIT;
 - ROLLBACK;
 - SAVEPOINT my_savepoint;
 - ROLLBACK TO my_savepoint;

Famous RDBMS



Data types supported by MySQL

- Mainly classified into three types
 - String data types
 - Numeric data types
 - Date and time data types
- Having data types can give us the following benefits
 - We can know the type of values it can have
 - The storage space the value can take
 - The value can be indexed or not
 - How comparison is performed between values of particular types

Data Type Syntax	Description
TINYINT	It is a very small integer that can be signed or unsigned. If signed, the allowable range is from -128 to 127. If unsigned, the allowable range is from 0 to 255. We can specify a width of up to 4 digits. It takes 1 byte for storage.
SMALLINT	It is a small integer that can be signed or unsigned. If signed, the allowable range is from -32768 to 32767. If unsigned, the allowable range is from 0 to 65535. We can specify a width of up to 5 digits. It requires 2 bytes for storage.
MEDIUMINT	It is a medium-sized integer that can be signed or unsigned. If signed, the allowable range is from -8388608 to 8388607. If unsigned, the allowable range is from 0 to 16777215. We can specify a width of up to 9 digits. It requires 3 bytes for storage.
INT	It is a normal-sized integer that can be signed or unsigned. If signed, the allowable range is from -2147483648 to 2147483647. If unsigned, the allowable range is from 0 to 4294967295. We can specify a width of up to 11 digits. It requires 4 bytes for storage.
BIGINT	It is a large integer that can be signed or unsigned. If signed, the allowable range is from -9223372036854775808 to 9223372036854775807. If unsigned, the allowable range is from 0 to 18446744073709551615. We can specify a width of up to 20 digits. It requires 8 bytes for storage.
FLOAT(m,d)	It is a floating-point number that cannot be unsigned. You can define the display length (m) and the number of decimals (d). This is not required and will default to 10,2, where 2 is the number of decimals, and 10 is the total number of digits (including decimals). Decimal precision can go to 24 places for a float type. It requires 2 bytes for storage.
DOUBLE(m,d)	It is a double-precision floating-point number that cannot be unsigned. You can define the display length (m) and the number of decimals (d). This is not required and will default to 16,4, where 4 is the number of decimals. Decimal precision can go to 53 places for a double. Real is a synonym for double. It requires 8 bytes for storage.
DECIMAL(m,d)	An unpacked floating-point number that cannot be unsigned. In unpacked decimals, each decimal corresponds to one byte. Defining the display length (m) and the number of decimals (d) is required. Numeric is a synonym for decimal.

BIT(m)	It is used for storing bit values into the table column. Here, M determines the number of bit per value that has a range of 1 to 64.
BOOL	It is used only for the true and false condition. It considered numeric value 1 as true and 0 as false.
BOOLEAN	It is Similar to the BOOL.

Date data types

Data Type Syntax	Maximum Size	Explanation
YEAR[(2 4)]	Year value as 2 digits or 4 digits.	The default is 4 digits. It takes 1 byte for storage.
DATE	Values range from '1000-01-01' to '9999-12-31'.	Displayed as 'yyyy-mm-dd'. It takes 3 bytes for storage.
TIME	Values range from '-838:59:59' to '838:59:59'.	Displayed as 'HH:MM:SS'. It takes 3 bytes plus fractional seconds for storage.
DATETIME	Values range from '1000-01-01 00:00:00' to '9999-12-31 23:59:59'.	Displayed as 'yyyy-mm-dd hh:mm:ss'. It takes 5 bytes plus fractional seconds for storage.
TIMESTAMP(m)	Values range from '1970-01-01 00:00:01' UTC to '2038-01-19 03:14:07' TC.	Displayed as 'YYYY-MM-DD HH:MM:SS'. It takes 4 bytes plus fractional seconds for storage.

Data Type Syntax	Maximum Size	Explanation
CHAR(size)	It can have a maximum size of 255 characters.	Here size is the number of characters to store. Fixed-length strings. Space padded on the right to equal size characters.
VARCHAR(size)	It can have a maximum size of 255 characters.	Here size is the number of characters to store. Variable-length string.
TINYTEXT(size)	It can have a maximum size of 255 characters.	Here size is the number of characters to store.
TEXT(size)	Maximum size of 65,535 characters.	Here size is the number of characters to store.
MEDIUMTEXT(size)	It can have a maximum size of 16,777,215 characters.	Here size is the number of characters to store.
LONGTEXT(size)	It can have a maximum size of 4GB or 4,294,967,295 characters.	Here size is the number of characters to store.
BINARY(size)	It can have a maximum size of 255 characters.	Here size is the number of binary characters to store. Fixed-length strings. Space padded on the right to equal size characters. (introduced in MySQL 4.1.2)
VARBINARY(size)	It can have a maximum size of 255 characters.	Here size is the number of characters to store. Variable-length string. (introduced in MySQL 4.1.2)
ENUM	It takes 1 or 2 bytes that depend on the number of enumeration values. An ENUM can have a maximum of 65,535 values.	It is short for enumeration, which means that each column may have one of the specified possible values. It uses numeric indexes (1, 2, 3...) to represent string values.
SET	It takes 1, 2, 3, 4, or 8 bytes that depends on the number of set members. It can store a maximum of 64 members.	It can hold zero or more, or any number of string values. They must be chosen from a predefined list of values specified during table creation.

Blob data types

Data Type Syntax	Maximum Size
TINYBLOB	It can hold a maximum size of 255 bytes.
BLOB(size)	It can hold the maximum size of 65,535 bytes.
MEDIUMBLOB	It can hold the maximum size of 16,777,215 bytes.
LOB	It can hold the maximum size of 4gb or 4,294,967,295 bytes.

Spatial data types

Data Types	Descriptions
GEOMETRY	It is a point or aggregate of points that can hold spatial values of any type that has a location.
POINT	A point in geometry represents a single location. It stores the values of X, Y coordinates.
POLYGON	It is a planar surface that represents multisided geometry. It can be defined by zero or more interior boundary and only one exterior boundary.
LINESTRING	It is a curve that has one or more point values. If it contains only two points, it always represents Line.
GEOMETRYCOLLECTION	It is a kind of geometry that has a collection of zero or more geometry values.
MULTILINESTRING	It is a multi-curve geometry that has a collection of linestring values.
MULTIPOINT	It is a collection of multiple point elements. Here, the point cannot be connected or ordered in any way.
MULTIPOLYGON	It is a multisurface object that represents a collection of multiple polygon elements. It is a type of two-dimensional geometry.

JSON Data Type

- Validates that the json syntax is correct
- Allows for first class searching
- Provides optimal storage format

MySQL Variables

- Used for storing data
- Labelling data with more appropriate name, readable
- Store data in memory and use throughout the program
- Three different types of mysql variables
 - User-defined variables
 - Local variables
 - System variables

User defined variables

- Pass some value from one statement to another statement
- SET and SELECT statement
- Starts with @symbol
- They are not case sensitive
- A user variable defined by one person cannot be used by other person
- SET @var_name = value; or SET @var_name: = value;
- SELECT @var_name;
- Accessing undeclared value gives null output

Local variable

- Not prefixed by @symbol
- It's a strongly typed variable
- Its scope is in stored program block only
- We use DECLARE keyword to specify the local variable
- We can also use DEFAULT clause to provide a default value
- Stores NULL value if the default value is not given
- DECLARE variable_name datatype(size) [DEFAULT default_value];
 - DECLARE a,b,c INT DEFAULT 0;
- We will look at examples while writing stored procedures/functions.

System variable

- Predefined variables
- Used by MySQL for its operations and configurations
- We can also change these values using the SET command
- These have two @@symbol before their names
- We can see all variables using
 - SHOW VARIABLES;
- See one single variable using @@variable_name;
 - SELECT @@host_cache_size;



[Connect with me on LinkedIn](#)



[Please subscribe to our YouTube channel](#)



[Check out my GitHub profile](#)



[Follow me on Twitter\(X\)](#)

Thank you